**IN THE SPECIFICATION:**

Please replace paragraph [0010] with the following amended paragraph:

**[0010]** As noted in the background section, conventional generational garbage collectors promote all objects in the Young Generation in accordance with a single policy. This results in premature promotion of some objects that become garbage after they have been moved to the next generation.

Please replace paragraph [0027] with the following amended paragraph:

**[0027]** As will be appreciated, the smart memory allocation system 102 can selectively delay or expedite promotion of an object to the next generation. Referring to Fig. 1, the smart memory allocation system 102 provides two modes of allocation, namely mode M1 and mode M2. Mode M1 can be considered a "normal" (or general) mode of allocation where objects are allocated in the Young Generation 108 so that they will be promoted in accordance with a normal threshold (e.g., surviving a number of garbage collection cycles). Mode M2, however, represents a "preemption" mode where objects are allocated in the Young Generation 108 in accordance with a policy that preempts the normal promotion policy. In other words, when the smart allocation system 102 uses mode M2 to allocate an object in the Young Generation 108, that normal promotion policy will not be enforced for these objects. Instead, these objects will be ~~prompted~~ promoted in accordance with a policy that is considered to be a better promotion policy for that object. As such, allocation mode M2 (preemption mode) effectively preempts the normal promotion policy that is enforced for objects allocated under mode M1 (normal mode). Typically, the preemption policy enforced under allocation mode M2 delays or avoids promotion of an object beyond the normal promotion threshold. As will be appreciated, it is desirable to

delay promotion of an object that is likely to become garbage within an acceptable time frame (e.g., before there is a need to expand the Young Generation). However, it should be noted that it is possible to use M2 (or another mode M3) to allocate an object that is effectively promoted to the next generation before the normal promotion threshold. This object may, for example, represent an object that is likely to be used well beyond the normal promotion threshold (e.g., permanent or long-lived objects).

Please replace paragraph [0035] with the following amended paragraph:

**[0035]** By way of example, while a code portion 308 is being executed, the allocation interface 302 may be in position S1 and switched to the general memory allocation 304. This means that a general allocation policy can be enforced. As a result, objects allocated with the general allocator 304 will be promoted in accordance with a general policy. However, before the code portion 310 is executed, a switch function can be called to change the allocation policy. In effect, when the code portion 310 is executed, the allocation switch is moved from the S1 to S2 position to inactivate the general memory allocator 304 and activate the preemptive memory allocator 306. As a result, all functions in the code portion 310 will use the preemptive memory allocator 306 even though the same allocation interface 302 is used. Hence, a function that may be nested or called by ~~anther~~ another function in the code portion 310 will allocate memory using the preemptive memory allocator 306.

Please replace paragraph [0037] with the following amended paragraph:

**[0037]** In other cases, however, it may be desirable to have a particular function directly use the preemptive memory allocator 306. Fig. 3B illustrates a preemptive memory allocator 350 that is

directly accessed (e.g., called) by a function in a code portion 352 in accordance with one embodiment of the invention. As shown in Fig. 3B, one or more functions in the code portion 352 may directly call the preemptive memory allocator 350 rather than a general memory allocator 354 that is used by other functions in the same code portion 352.

Please replace paragraph [0042] with the following amended paragraph:

**[0042]** Fig. 5 depicts a promotion method 500 for promoting a live object which has been allocated in one generation to the next generation of a memory portion in accordance with one embodiment of the invention. Typically, the object promotion method 500 is performed at each garbage collection cycle of a generational garbage collection scheme. During each garbage collection cycle, objects that may still be used (are alive) are examined to determine whether they should be promoted from the Young Generation to the next generation. To provide a more comprehensive demonstration, the object promotion method 500 illustrates a promotion method that can be used for various header embodiments shown in Fig. 2C, 2D and 2E. However, it should be noted that it is possible to use a different promotion method for each of these embodiments. As such, a subset of the operations may be used, for example, in cases when only one of the header types is used.

Please replace paragraph [0043] with the following amended paragraph:

**[0043]** Initially, a Garbage Collection Count (GCC) is read 502 for the object. The Garbage Collection Count is a positive integer which has been initialized for the object when the object was allocated. The Garbage Collection Count (GCC) reflects the promotion policy that has been

selected for the object. In this example, larger[[s]] integer values, correspond to longer waiting periods.

Please replace paragraph [0046] with the following amended paragraph:

**[0046]** On the other hand, if it is determined 508 that a preemption indicator is set, it is determined 512 whether a preemption value (e.g., 220 of Fig. 2D) has been provided. If it is determined 512 a preemption value has not been provided, the object promotion method 500 ends. In other words, the object is not promoted to the next generation even though the GCC count is zero. In effect, the GCC is ignored when the preemptive indicator is set, but no preemptive value has been provided (e.g., header 203 shown in Fig. 2C). However, when it is determined 512 that a preemption value has been provided, the preemptive value is read 514. Accordingly, it is determined 516 whether the preemptive value is equal to zero. If the preemptive value is greater than zero, it is decreased 518 [[be]] by one and the object promotion method 500 ends. It should be noted that the object is not promoted to the next generation until it is determined 516 that the preemptive value is equal to zero. Accordingly, if determined 516 that the preemptive value is equal to zero, the object is promoted 510 to the next generation, and the object promotion method 500 ends.